

KNN for Text Summarization based on Text Categorization considering Feature Similarity

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the KNN version which considers the feature similarities, for automating the text summarization by combining it with the text categorization. In the previous work, we proposed the modernized KNN version as an approach to the text summarization, but must tag the input text with its domain, manually. In this research, the KNN algorithm is modified into the version which considers the feature similarities and applied to the text summarization and the text classification. In the proposed system, only a text is given as the input, blinding its domain, it is assigned to the text by the text classification, and the summary is extracted by the classifier which corresponds to the domain. The goals of this research are to automate the text summarization by introducing the text classification and to improve the performance of both tasks, using the modernized KNN algorithm.

I. INTRODUCTION

The text summarization is the process of extracting the essential part from a text as its summary. The essential part is assumed to be some paragraphs in the text, so the text summarization is mapped into a binary classification of each paragraph into summary or non-summary. The process of text summarization is to partition a text into paragraphs, to classify them into one of the two categories, and extract ones which are classified into summary. The binary classification which is mapped from the text summarization is a domain dependent classification where a same entity is classified differently depending on the domain. It is required to present the domain as a tag for summarizing a text.

This research is motivated by the requirement of presenting a domain for executing the text summarization. It is set as an independent binary classification domain by domain. We need to know the domain of the text for selecting a classifier which is consistent with it. It is possible to execute the text summarization with only a text by assigning the domain automatically to it through the text classification. The text summarization which is covered in this research relates to the text classification.

In this research, we connect the text summarization with the text classification and apply the KNN algorithm which considers the feature similarities to both tasks. We define the similarity metric between numerical vectors and adopt it for computing the similarities of each novice item with

the training examples in the KNN algorithm. The modified KNN version is applied to the text classification which automatically decides a domain. The input text is partitioned into paragraphs, the modified KNN version is applied to the classification of paragraphs into summary or non-summary, and ones which are classified into summary are extracted as the final output. The modified KNN version is adopted for implementing both the text summarization and the text classification.

Let us mention some benefits from this research. The similarity metric between two numerical vectors which considers the feature similarities is defined as the solution to the poor discrimination among sparse vectors. The performance of the text summarization and the text classification is improved by adopting the KNN algorithm which considers the feature similarities as the approach to both tasks. Presenting a domain as a tag manually for the text summarization system is avoided by connecting the text summarization with the text classifier. We automate selecting a classifier which corresponds to a domain in the text summarization system as its upgrade.

Let us mention the remaining tasks for upgrading this research. Only some features are allowed in computing the feature similarities for reducing the computation time. Paragraphs and texts may be represented into compound structured forms, each of which consists of more than one structured form. Other machine learning algorithms and deep learning algorithms are modified into the proposed versions. The text summarization system may be implemented as a real program or a library by adopting the proposed KNN version.

II. PROPOSED WORKS

A. Encoding Process

This section is concerned with the process of encoding a text into a numerical vector. Words are used as features for encoding a text so, and some are selected from words in a corpus. The similarity between words is computed based on texts with their collocations and is used as the similarity between features. The similarities among features are considered for computing the semantic similarity between texts.

In this section, we describe the process of encoding a text into a numerical vector and the process of computing the similarity between texts.

The process of encoding texts into numerical vectors is illustrated in Figure 1. The entire text collection is indexed into a list of words which are feature candidates. Some words are selected as the features by criteria among the feature candidates in the second step. The weights of the selected features in the text are computed as their relationships with the text, as elements of the numerical vector. Refer to [1] for studying the process and the selection criteria in detail.

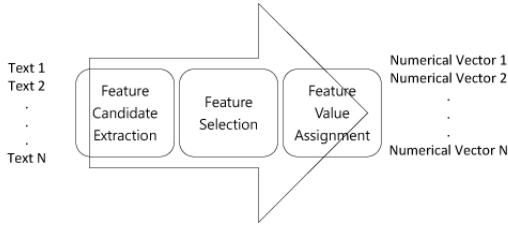


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_i, f_j)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

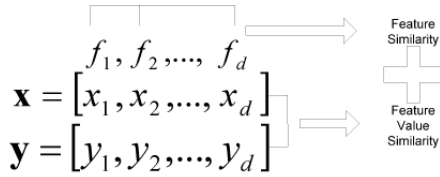


Figure 2. Frame of Computing Similarity between Numerical Vectors

Let us mention the process of computing the similarity between numerical vectors as the similarity between texts. The similarity between the features, f_i and f_j , is notated by $\text{sim}(f_i, f_j) = s_{ij}$. The similarity between the features is computed by equation (1),

$$s_{ij} = \text{sim}(t_i, t_j) = \frac{2 \times \#(t_i \wedge t_j)}{\#(t_i) + \#(t_j)}, \quad (1)$$

where $\#(t_i \wedge t_j)$, is the number of texts which include both words, t_i and t_j , $\#(t_i)$ is the number of texts which include the word, t_i , and $\#(t_j)$ is the number of texts which include the word, t_j , and the similarity matrix to the d features is

constructed as a $d \times d$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

The similarity between the two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (2),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \|\mathbf{x}\| \|\mathbf{y}\|} \quad (2)$$

Equation (2) is used for computing the similarity between a novice text and a sample text in the KNN algorithm.

Let us make some remarks on the encoding process and the computation of the similarity between two texts. A text is encoded into a numerical vector by the feature candidate extraction, the feature extraction, and the feature value assignment. The similarity between features which are given as words is computed by equation (1). The similarity between numerical vectors which represent texts is computed by equation (2). In future, we consider computing the similarities among some features rather than all features for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text summarization by Jo in 2019 [2]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the initial version of the KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (2), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice

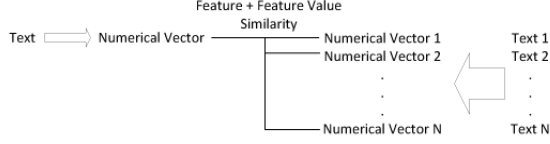


Figure 3. Similarities of Novice Input with Training Examples

example are selected as its nearest neighbors, as expressed in equation (3),

$$Nr = \text{Select}_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (3)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (4),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \quad (4)$$

The elements in the set, Nr , are used for voting their labels in the next step.

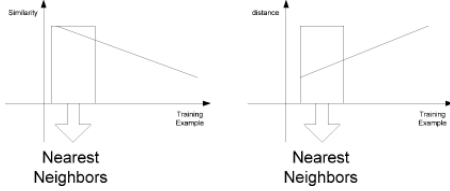


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (5),

$$\text{label}(\mathbf{x}) = \arg\max_{j=1}^m \text{Count}(Nr, c_j) \quad (5)$$

The process of deciding the label of a novice item by equation (5), is called voting.

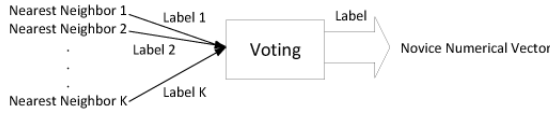


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the KNN algorithm which considers the feature similarities. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors.

The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

The process of sampling paragraphs domain by domain is illustrated in Figure 6. Even if it is possible map the text summarization into an instance of text classification, a paragraph may be classified differently depending on the domain. Sample paragraphs which are labeled with summary or non-summary are gathered domain by domain. The text which is given as the input is tagged with a domain, and the classifier which corresponds to the domain is appointed. The sample paragraphs are encoded into numerical vectors in each domain.

C. System Architecture

This section is concerned with the text summarization system which adopts the KNN algorithm which considers the feature similarities. In Section II-B, we described the proposed version of KNN algorithm as the approach to the text summarization. It is viewed as a binary classification which classifies a paragraph into summary or non-summary. This section is intended to describe the text summarization system with respect to its function and its architecture.

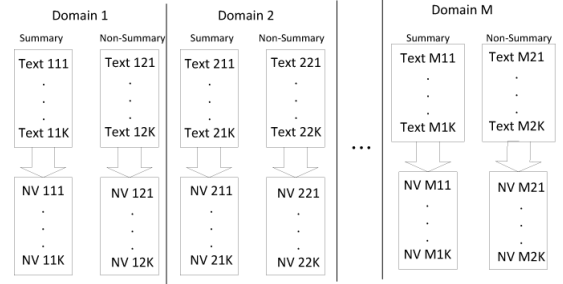


Figure 6. Preparation of Training Examples

The system architecture of the text summarization system is illustrated in Figure 7. The text partition module partitions the text which is given as the input into paragraphs. The encoder module encodes the paragraphs into numerical vectors by the process which is described in Section II-A. The similarity computation module computes the similarities of each paragraph with the sample paragraphs by the process which is described in Section II-A and selects ones with highest similarities as the nearest neighbors. The voting module votes the labels of nearest neighbors for decoding the label of the novice item.

The execution flow of the text summarization system is illustrated in Figure 8. In each domain, sample paragraphs which are labeled with summary or non-summary are gathered. The input text is partitioned into novice paragraphs, and both sample paragraphs and novice paragraphs are encoded into numerical vectors. Each paragraph is classified into summary or non-summary, and there are two groups of paragraphs in the text: summary group and non-summary

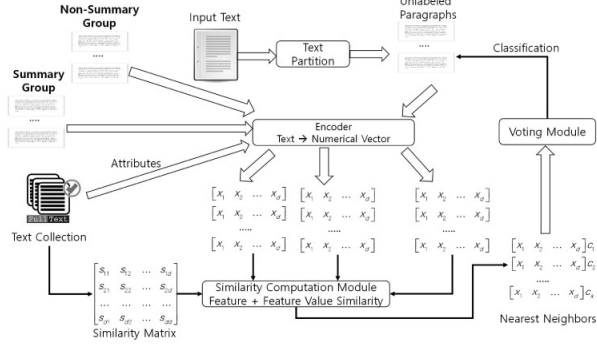


Figure 7. System Architecture

group. The paragraph which are classified into summary are extracted as the summary of the input text in this system.

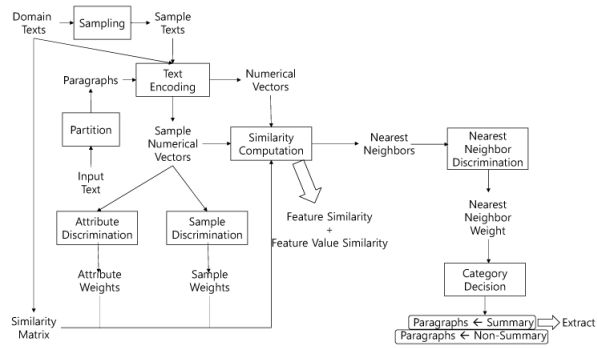


Figure 8. System Flow

Let us make some remarks on the system architecture and the execution process of the text summarization system which are presented in Figure 7 and 8. The text summarization is viewed into a binary classification of paragraphs, and we propose the similarity metric between two numerical vectors which is tolerant to the sparse distribution over them. The KNN algorithm is modified by defining the similarity metric between a novice paragraph and a sample paragraph, and the proposed version is applied to the text summarization. The system architecture and the execution flow which are provided by this research are needed for making the general design of the system. In the next research, we consider the detail design and the implementation of the system.

D. Connection with Text Classification

This section is concerned with the connection of the text summarization with the text classification. In the previous section, we mentioned the text summarization system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for

implementing the text classification. This section is intended to describe the connection of the text summarization system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into numerical vectors by the process which is described in Section II-A. The similarity computation module is for computing the similarities between numerical vectors which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system is used for automating deciding the domain of the input text.

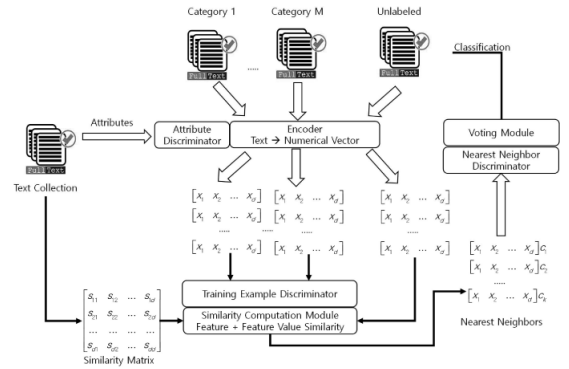


Figure 9. Text Categorization System

The connection of the text summarization with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the text summarization system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the text summarization, automatically.

The process of executing the text summarization, connecting it with the text classification is illustrated in Figure 11. The sample texts each of which is labeled with its own domain are prepared, and the sample paragraphs each of which is labeled with summary or non-summary are prepared in each domain. A novice text is classified into one among the predefined domains, and the classifier which corresponds to the domain is nominated. The novice text is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. The paragraphs which are labeled with summary is the final output from this system; the domain which is assigned to the input text is the

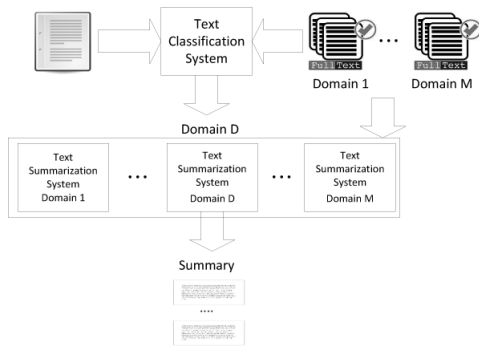


Figure 10. Text Summarization System connected with Text categorization

guide for selecting a classifier.

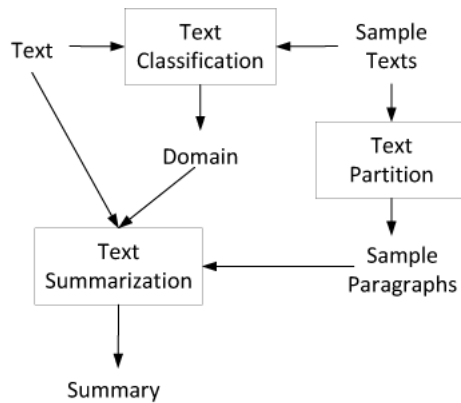


Figure 11. System Flow of Text Summarization connected with Text Categorization

Let us make some remarks on the connection of the text summarization with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the text summarization is to extract important paragraphs as the summary. In the execution flow, the input text is classified into a domain, it is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. We consider connecting the text classification as the main task the text summarization as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Text Mining: Concepts and Big Data Challenge", Springer, 2019.
- [2] T. Jo, "text summarization using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 2019.